LETTER **Communicated by Alessandro Treves**

Noise Facilitation in Associative Memories of Exponential Capacity

Amin Karbasi
amin.karbasi@yale.edu
Yale University, New Haven, CT 06511, U.S.A.

Amir Hesam Salavati
hesam.salavati@epfl.ch
Amin Shokrollahi
amin.shokrollahi@epfl.ch
Ecole Polytechnique Federale de Lausanne, Lausanne 1015, Switzerland

Lav R. Varshney
varshney@illinois.edu
University of Illinois at Urbana-Champaign, Urbana, IL 61801, U.S.A.

Recent advances in associative memory design through structured pattern sets and graph-based inference algorithms have allowed reliable learning and recall of an exponential number of patterns that satisfy certain subspace constraints. Although these designs correct external errors in recall, they assume neurons that compute noiselessly, in contrast to the highly variable neurons in brain regions thought to operate associatively, such as hippocampus and olfactory cortex. Here we consider associative memories with boundedly noisy internal computations and analytically characterize performance. As long as the internal noise level is below a specified threshold, the error probability in the recall phase can be made exceedingly small. More surprising, we show that internal noise improves the performance of the recall phase while the pattern retrieval capacity remains intact: the number of stored patterns does not reduce with noise (up to a threshold). Computational experiments lend additional support to our theoretical analysis. This work suggests a functional benefit to noisy neurons in biological neuronal networks.

1 Introduction

Brain regions such as hippocampus and olfactory cortex are thought to operate as associative memories (Treves & Rolls, 1994; Stettler & Axel, 2009; Wilson & Sullivan, 2011), having the ability to learn patterns from presented inputs, store a large number of patterns, and retrieve them reliably

Neural Computation **26, 2493–2526** (2014) © 2014 Massachusetts Institute of Technology
doi:10.1162/NECO_a_00655

in the face of noisy queries (Hopfield, 1982; McEliece, Posner, Rodemich, & Venkatesh, 1987; Amit & Fusi, 1994). Mathematical models of associative memory are therefore designed to memorize a set of given patterns so that corrupted versions of the memorized patterns may later be presented and the correct memorized pattern retrieved.

Although such information storage and recall seemingly falls naturally into the information-theoretic framework (Palm, 1980), where an exponential number of messages can be communicated reliably using a linear number of symbols (Shannon, 1948), classical associative memory models can store only a linear number of patterns with a linear number of symbols (McEliece et al., 1987). A primary shortcoming of such classical models has been their requirement to memorize a randomly chosen set of patterns. By enforcing structure and redundancy in the possible set of memorizable patterns through subspace constraints—much like natural stimuli (Olshausen and Field, 2004), internal representations in neural systems (Koulakov & Rinberg, 2011), and code words in error-control codes (Richardson & Urbanke, 2008)—new advances in associative memory design allow storage of an exponential number of patterns with a linear number of symbols (Salavati & Karbasi, 2012; Karbasi, Salavati, & Shokrollahi, 2013), just like in communication systems.[1]

Information-theoretic and associative memory models of storage have been used to predict experimentally measurable properties of synapses in the mammalian brain (Brunel, Hakim, Isope, Nadal, & Barbour, 2004; Varshney, Sjöström, & Chklovskii, 2006). But contrary to the fact that noise is present in computational operations of the brain (Koch, 1999; Faisal, Selen, & Wolpert, 2008; Rolls & Deco, 2010; McDonnell & Ward, 2011; Destexhe & Rudolph-Lilith, 2012), associative memory models with exponential capacity have assumed no internal noise in the computational nodes (Karbasi et al., 2013); the same is true with many classical models (Hopfield, 1982). The purpose of this letter is to model internal noise in associative memories with exponential pattern retrieval capacity and study whether they are still able to operate reliably. A particular model of neuronal computation and of bounded internal noise is assumed. Surprisingly, we find that internal noise actually enhances recall performance without loss in capacity, thereby suggesting a functional role for variability in the brain.

In particular, we consider a convolutional, graph code–based, associative memory model (Karbasi et al., 2013) and find that even if all components are noisy, the final error probability in recall can be made exceedingly small. We characterize a threshold phenomenon and show how to optimize

[1] The idea of restricted pattern sets leading to associative memories with increased storage capacity was first suggested in an unpublished doctoral dissertation (Biswas, 1993).

algorithm parameters when knowing statistical properties of internal noise. Rather counterintuitively, the performance of the memory model improves in the presence of internal neural noise, as has been observed previously as stochastic resonance in the literature (Chen, Varshney, Kay, & Michels, 2007; McDonnell & Ward, 2011). Deeper analysis shows mathematical connections to perturbed simplex algorithms for linear programing (Spielman & Teng, 2004), where some internal noise helps the algorithm get out of local minima.

1.1 Related Work. Designing neural networks to learn a set of patterns and recall them later in the presence of noise has been an active topic of research for the past three decades. Inspired by Hebbian learning (Hebb, 1949), Hopfield (1982) introduced an autoassociative neural mechanism of size n with binary state neurons in which patterns are assumed to be binary vectors of length n. The capacity of a Hopfield network under vanishing block error probability was later shown to be $O(n/\log(n))$ (McEliece et al., 1987). With the hope of increasing the capacity of the Hopfield network, extensions to nonbinary states were explored (Amit & Fusi, 1994). In particular, Jankowski, Lozowski, and Zurada (1996) investigated a multistate complex-valued neural associative memory with estimated capacity less than $0.15n$; Müezzinoğlu, Güzeliş, and Zurada (2003) showed the capacity with a prohibitively complicated learning rule to increase to n. Lee (2006) proposed the modified gradient descent learning rule (MGDR) to overcome this drawback.

To increase capacity and robustness further, a recent line of work considers exploiting structure in patterns. This is done either by making use of correlations among patterns or by memorizing only patterns with redundancy (rather than any possible set of patterns). By utilizing neural cliques, Gripon and Berrou (2011) demonstrated that increasing the pattern retrieval capacity of Hopfield networks to $O(n^2)$ is possible. Modification of neural architecture to improve pattern retrieval capacity has also been previously considered by Venkatesh (1994) and Biswas (1993), where the capacity is increased to $\Theta\left(b^{n/b}\right)$ for semirandom patterns, where $b = \omega(\ln n)$ is the size of clusters. This significant boost to capacity is achieved by dividing the neural network into smaller fully interconnected disjoint blocks or nested blocks (Baram, 1991). This huge improvement comes at the price of limited worst-case noise tolerance capabilities. Deploying higher-order neural models beyond the pairwise correlation considered in Hopfield networks increases the storage capacity to $O(n^{p-2})$, where p is the degree of correlation (Peretto & Niez, 1986). In such models, neuronal state depends not only on the state of neighbors, but also on the correlations among them. A new model based on bipartite graphs that captures higher-order correlations (when patterns belong to a subspace), but without prohibitive computational complexity, improved capacity to $O(a^n)$ for some $a > 1$, that is, *exponential* in network size (Salavati & Karbasi, 2012).

The basic memory architecture, learning rule, and recall algorithm used here is from Karbasi et al. (2013), which also achieves exponential capacity by capturing internal redundancy by dividing the patterns into smaller overlapping clusters, with each subpattern satisfying a set of linear constraints. The problem of learning linear constraints with neural networks was considered in Xu, Krzyzak, and Oja (1991), but without sparsity requirements. This has connections to compressed sensing (Candés & Tao, 2006); typical compressed sensing recall and decoding algorithms are too complicated to be implemented by neural networks, but some have suggested the biological plausibility of message-passing algorithms (Fletcher, Rangan, Varshney, & Bhargava, 2011).

Building on the idea of structured pattern sets (Gripon & Berrou, 2011), the basic associative memory model used here (Karbasi et al., 2013) relies on the fact that all patterns to be learned lie in a low-dimensional subspace. Learning features of a low-dimensional space are very similar to autoencoders (Vincent, Larochelle, Bengio, & Manzagol, 2008). The model also has similarities to deep belief networks (DBNs) and, in particular, convolutional neural networks (Le et al., 2010), albeit with different objectives. DBNs are made of several consecutive stages, similar to overlapping clusters in our model, where each stage extracts some features and feeds them to the following stage. The output of the last stage is then used for pattern classification. In contrast to DBNs, our associative memory model is not classifying patterns but rather is recalling patterns from noisy versions. Also, overlapping clusters can operate in parallel to save time in information diffusion over a staged architecture.

In most deep or convolutional models, one not only has to find the proper dictionary for classification, but also calculate the features for each input pattern. This increases the complexity of the whole system when the objective is simply recall. Here the dictionary corresponds to the dual vectors from previously memorized patterns.

In this work, we reconsider the neural network model of Karbasi et al. (2013) but introduce internal computation noise consistent with biology. Note that the sparsity of the model architecture is also consistent with biology (Song, Sjöström, Reigl, Nelson, & Chklovskii, 2005). We find that there is actually a functional benefit to internal noise.

The benefit of internal noise has been noted previously in associative memory models with stochastic update rules (Amit, 1992) by analyzing attractor dynamics. In particular, it has been shown that noise may reduce recall time in associative memory tasks by pushing the system from one attractor state to another (Liljenström & Wu, 1995). However, our framework differs from previous approaches in three key aspects. First, our memory model is different, which makes extension of previous analysis nontrivial. Second, and perhaps most important, pattern retrieval capacity in previous approaches decreases with internal noise (Amit, 1992, Figure 6.1), in that increasing internal noise helps correct more external errors but also reduces the number of memorizable patterns. In our framework, internal noise does not affect pattern retrieval capacity (up to a threshold) but improves recall performance. Finally, our noise model has bounded rather than gaussian noise, and so a suitable network may achieve perfect recall despite internal noise.

Reliably storing information in memory systems constructed completely from unreliable components is a classical problem in fault-tolerant computing (Taylor, 1968; Kuznetsov, 1973; Varshney, 2011), where typical models have used random access architectures with sequential correcting networks. Although direct comparison is difficult since notions of circuit complexity are slightly different, our work also demonstrates that associative memory architectures can store information reliably despite being constructed from unreliable components.

2 Associative Memory Model

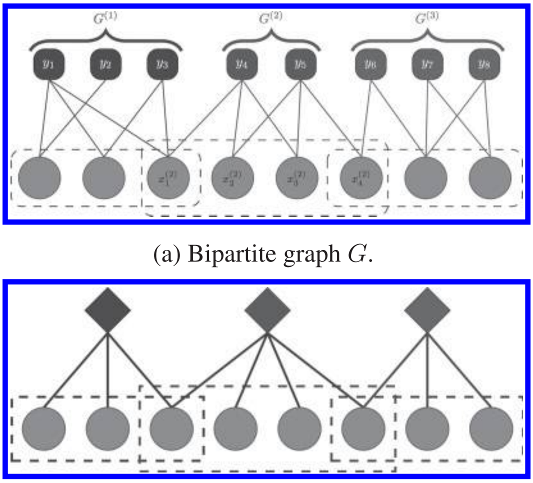
In this section, we introduce our main notation, the model of associative memories and noise. We also explain the recall algorithms.

2.1 Notation and Basic Structure. In our model, a neuron can assume an integer-valued state from the set $\mathcal{Q} = \{0, \ldots, Q-1\}$, interpreted as the short-term firing rate of neurons. A neuron updates its state based on the states of its neighbor $\{s_i\}_{i=1}^n$ as follows. It first computes a weighted sum $h = \sum_{i=1}^n w_i s_i + \zeta$, where w_i is the weight of the link from s_i and ζ is the internal noise, and then applies nonlinear function $f : \mathbb{R} \rightarrow \mathcal{Q}$ to h.

An associative memory is represented by a weighted bipartite graph, G, with pattern neurons and constraint neurons. Each pattern $x = (x_1, x_2, \ldots, x_n)$ is a vector of length n, where $x_i \in \mathcal{Q}$, $i = 1, \ldots, n$. Following Karbasi et al. (2013), the focus is on recalling patterns with strong local correlation among entries. Hence, we divide entries of each pattern x into L overlapping subpatterns of lengths $n_1, n_2, \ldots, n_L$. Due to overlaps, a pattern neuron can be a member of multiple subpatterns, as depicted in Figure 1a. The ith subpattern is denoted $x^{(i)} = (x_1^{(i)}, x_2^{(i)}, \ldots, x_{n_i}^{(i)})$, and local correlations are assumed to be in the form of subspaces, that is, the subpatterns $x^{(i)}$ form a subspace of dimension $k_i < n_i$.

We capture the local correlations by learning a set of linear constraints over each subspace corresponding to the dual vectors orthogonal to that subspace. More specifically, let $\{w_1^{(i)}, w_2^{(i)}, \ldots, w_{m_i}^{(i)}\}$ be a set of dual vectors orthogonal to all subpatterns $x^{(i)}$ of cluster i. Then:

$$y_j^{(i)} = \left(w_j^{(i)}\right)^T \cdot x^{(i)} = 0, \quad \text{for all } j \in \{1, \ldots, m_i\} \text{ and for all } i \in \{1, \ldots, L\}. \tag{2.1}$$



(a) Bipartite graph G.

(b) Contraction graph $\widetilde{G}$.

Figure 1: The proposed neural associative memory with overlapping clusters.

Equation 2.1 can be rewritten as $W^{(i)} \cdot x^{(i)} = 0$, where $W^{(i)} = [w_1^{(i)}|w_2^{(i)}|\cdots|w_{m_i}^{(i)}]^T$ is the matrix of dual vectors. Now we use a bipartite graph with connectivity matrix determined by $W^{(i)}$ to represent the subspace constraints learned from subpattern $x^{(i)}$; this graph is called *cluster i*. We developed an efficient way of learning $W^{(i)}$ in Karbasi et al. (2013), also used here. Briefly, in each iteration of learning:

1. Pick a pattern x at random from the data set.
2. Adjust weight vectors $w_j^{(i)}$ for $j = \{1, \ldots, m_i\}$ and $i = \{1, \ldots, L\}$ such that the projection of x onto $w_j^{(i)}$ is reduced. Apply a sparsity penalty to favor sparse solutions.

This process repeats until all weights are orthogonal to the patterns in the data set or the maximum iteration limit is reached. The learning rule allows us to assume the weight matrices $W^{(i)}$ are known and satisfy $W^{(i)} \cdot x^{(i)} = 0$ for all patterns x in the data set $\mathcal{X}$, in this letter.

For the asymptotic analysis, we need to define a contracted graph $\widetilde{G}$ whose connectivity matrix is denoted $\widetilde{W}$ and has size $L \times n$. This is a bipartite graph in which constraints in each cluster are represented by a single neuron. Thus, if pattern neuron x_j is connected to cluster i, $\widetilde{W}_{ij} = 1$; otherwise $\widetilde{W}_{ij} = 0$ (see Figure 1b). We also define the degree distribution from an edge perspective over $\widetilde{G}$, using

$$\widetilde{\lambda}(z) = \sum_j \widetilde{\lambda}_j z^{j-1}, \tag{2.2}$$

$$\widetilde{\rho}(z) = \sum_j \widetilde{\rho}_j z^{j-1}, \tag{2.3}$$

where $\widetilde{\lambda}_j$ (resp., $\widetilde{\rho}_j$) equals the fraction of edges that connect to pattern (resp., cluster) nodes of degree j.

2.2 Noise Model. There are two types of noise in our model: external errors and internal noise. A neural network should be able to retrieve memorized pattern $\hat{x}$ from its corrupted version x due to external errors. We assume the external error is an additive vector of size n, denoted by z satisfying $x = \hat{x} + z$, whose entries assume values independent from $\{-1, 0, +1\}$ with corresponding probabilities $p_{-1} = p_{+1} = \epsilon/2$ and $p_0 = 1 - \epsilon$.[2] The realization of the external error on subpattern $x^{(i)}$ is denoted $z^{(i)}$. Note that the subspace assumption implies $W \cdot y = W \cdot z$ and $W^{(i)} \cdot y^{(i)} = W^{(i)} \cdot z^{(i)}$ for all i.

Neurons also suffer from internal noise. We consider a bounded noise model—a random number uniformly distributed in the intervals $[-\upsilon, \upsilon]$ and $[-\nu, \nu]$ for the pattern and constraint neurons, respectively ($\upsilon, \nu < 1$).

The goal of recall is to filter the external error z to obtain the desired pattern x as the correct states of the pattern neurons. When neurons compute noiselessly, this task may be achieved by exploiting the fact that the set of patterns $x \in \mathcal{X}$ satisfies the set of constraints $W^{(i)} \cdot x^{(i)} = 0$. However, it is not clear how to accomplish this objective when the neural computations are noisy. Rather surprisingly, we show that eliminating external errors is not only possible in the presence of internal noise, but that neural networks with moderate internal noise demonstrate better external error resilience.

2.3 Recall Algorithms. To efficiently deal with external errors in associative memory, we use two simple iterative message-passing algorithms. The role of the first one, called the intracluster algorithm and formally defined in algorithm 1, is to correct at least a single external error in each cluster. However, without overlaps between clusters, the error resilience of this approach and the network in general is limited. The second algorithm, the intercluster recall algorithm, exploits the overlaps: it helps clusters with external errors recover their correct states by using the reliable information from clusters that do not have external errors. The error resilience of the resulting combination thereby drastically improves.

[2] Note that the proposed algorithms also work with larger noise values, that is, from a set $\{-S, \ldots, S\}$ for some $S \in \mathbb{N}$ (see section 4.2). The ±1 noise model is presented here for simplicity.

Algorithm 1: Intramodule Error Correction.

Input: Training set $\mathcal{X}$, thresholds φ, ψ, iteration $t_{\max}$

Output: $x_1^{(\ell)}, x_2^{(\ell)}, \ldots, x_{n_\ell}^{(\ell)}$

```
for $t = 1 \to t_{\max}$ do
    Forward iteration: Calculate the input $h_i^{(\ell)} = \sum_{j=1}^{n_\ell} W_{ij}^{(\ell)} x_j^{(\ell)} + v_i$, for each neuron $y_i^{(\ell)}$ and set $y_i^{(\ell)} = f(h_i^{(\ell)}, \psi)$.
    Backward iteration: Each neuron $x_j^{(\ell)}$ computes
        $g_j^{(\ell)} = \frac{\sum_{i=1}^{m_\ell} \text{sign}(W_{ij}^{(\ell)}) y_i^{(\ell)}}{\sum_{i=1}^{m_\ell} \text{sign}(|W_{ij}^{(\ell)}|)} + u_i.$
    Update the state of each pattern neuron $j$ according to $x_j^{(\ell)} = x_j^{(\ell)} - \text{sign}(g_j^{(\ell)})$ only if $|g_j^{(\ell)}| > \varphi$.
end for
```

To go further into details and with abuse of notations, let $x_i(t)$ and $y_j(t)$ denote the message transmitted at iteration t by pattern and constraint neurons, respectively. In the first iteration, we initialize the pattern neurons with a pattern randomly drawn from the data set, $\hat{x}$, corrupted with some external noise, z. Thus, $x(0) = \hat{x} + z$. As a result, for cluster ℓ, we have $x^{(\ell)}(0) = \hat{x}^{(\ell)} + z^{(\ell)}$, where $z^{(\ell)}$ is the realization of the external error on cluster ℓ.

With these notations in mind, algorithm 1 iteratively performs a series of forward and backward steps in order to remove (at least) one external error from its input domain. Assuming that the algorithm is applied to cluster ℓ, in the forward step of iteration t, the pattern neurons in cluster ℓ transmit their current states to their neighboring constraint neurons. Each constraint neuron j then calculates the weighted sum of the messages it received over its input links. Nevertheless, since neurons suffer from internal noise, additional noise terms appear in the weighted sum, $h_j^{(\ell)} = \sum_{i=1}^{n_\ell} W_{ij}^{(\ell)} x_i^{(\ell)} + v_i$, where v_i is the random internal noise affecting node i. As before, we consider a bounded noise model for v_i it is uniformly distributed in the interval $[-\nu, \nu]$ for some $\nu < 1$.

A nonzero input sum, excluding the effect of v_i, is an indication of the presence of external errors among the pattern neurons. Thus, constraint neurons set to their states to the sign of the received weighted sum if its magnitude is larger than a fixed threshold, ψ. More specifically, constraint neuron j updates its state based on the received weighted sum according to the following rule:

$$y_j^{(\ell)}(t) = f\big(h_j^{(\ell)}(t), \psi\big) = \begin{cases} +1, & \text{if } h_j^{(\ell)}(t) \geq \psi \\ 0, & \text{if } -\psi \leq h_j^{(\ell)}(t) \leq \psi. \\ -1, & \text{otherwise} \end{cases} \tag{2.4}$$

Here, $x^{(\ell)}(t) = [x_1^{(\ell)}(t), \ldots, x_{n_\ell}^{(\ell)}(t)]$ is the vector of messages transmitted by the pattern neurons and v_i is the random internal noise affecting node i.[3]

In the backward step, the constraint neurons communicate their states to their neighboring pattern neurons. The pattern neurons then compute a normalized weighted sum on the messages they receive over their input link and update their current state if the amount of received (nonzero) feedback exceeds a threshold. Otherwise they will retain their current state for the next round. More specifically, pattern node i in cluster ℓ updates its state in round t according to

$$x_i^{(\ell)}(t+1) = \begin{cases} x_i^{(\ell)}(t) - \operatorname{sign}\big(g_i^{(\ell)}(t)\big), & \text{if } \big|g_i^{(\ell)}(t)\big| \geq \varphi \\ x_i^{(\ell)}(t), & \text{otherwise} \end{cases}, \tag{2.5}$$

where φ is the update threshold and

$$g_i^{(\ell)}(t) = \frac{((\operatorname{sign}(W^{(\ell)})^\top \cdot y^{(\ell)}(t))_i}{d_i^{(\ell)}} + u_i.$$

Note that $x_i^{(\ell)}(t+1)$ is further mapped to the interval $[0, Q-1]$ by saturating the values below 0 and above $Q-1$ to 0 and $Q-1$, respectively; this saturation is not stated mathematically for brevity. Here, $d_i^{(\ell)}$ is the degree of pattern node i in cluster ℓ, $y^{(\ell)}(t) = [y_1^{(\ell)}(t), \ldots, y_{m_\ell}^{(\ell)}(t)]$ is the vector of messages transmitted by the constraint neurons in cluster ℓ, and u_i is the random internal noise affecting pattern node i. Basically, the term $g_i^{(\ell)}(t)$ reflects the (average) belief of constraint nodes connected to pattern neuron i about its correct value. If $g_i^{(\ell)}(t)$ is larger than a specified threshold φ, it means that most of the connected constraints suggest the current state $x_i^{(\ell)}(t)$ is not correct; hence, a change should be made. Note that this average belief is diluted by the internal noise of neuron i. As mentioned earlier, u_i is uniformly distributed in the interval $[-\upsilon, \upsilon]$, for some $\upsilon < 1$.

[3] Note that although the values of $y_i^{(\ell)}(t)$ can be shifted to $0, 1, 2$, instead of $-1, 0, 1$ to match our assumption that neural states are nonnegative, we leave them as such to simplify later analysis.

Algorithm 2: Sequential Peeling Algorithm.

Input: $\widetilde{G}, G^{(1)}, G^{(2)}, \dots, G^{(L)}$

Output: $x_1, x_2, \dots, x_n$

```
while there is an unsatisfied $v^{(\ell)}$ do
    for $\ell = 1 \to L$ do
        If $v^{(\ell)}$ is unsatisfied, apply algorithm 1 to cluster $G^{(l)}$.
        If $v^{(\ell)}$ remained unsatisfied, revert the state of pattern neurons connected to
        $v^{(\ell)}$ to their initial state. Otherwise, keep their current states.
    end for
end while
Declare $x_1, x_2, \dots, x_n$ if all $v^{(\ell)}$'s are satisfied. Otherwise, declare failure.
```

The error correction ability of algorithm 1 is fairly limited, as determined analytically and through simulations in the sequel. In essence, algorithm 1 can correct one external error with high probability but degrades terribly against two or more external errors. Working independently, clusters cannot correct more than a few external errors, but their combined performance is much better. As clusters overlap, they help each other in resolving external errors: a cluster whose pattern neurons are in their correct states can always provide truthful information to neighboring clusters. This property is exploited in algorithm 2 by applying algorithm 1 in a round-robin fashion to each cluster. Clusters either eliminate their internal noise, in which case they keep their new states and can now help other clusters, or revert to their original states. Note that by such a scheduling scheme, neurons can change their states only toward correct values. This scheduling technique is similar in spirit to the peeling algorithm (Luby, Mitzenmacher, Shokrollahi, & Spielman, 2001).

3 Pattern Retrieval Capacity

Before proceeding to analyze recall performance, for completeness we review pattern retrieval capacity results from Karbasi et al. (2013) to show that the proposed model is capable of memorizing an exponentially large number of patterns. First, note that since the patterns form a subspace, the number of patterns C does not have any effect on the learning or recall algorithms (except for its obvious influence on the learning time). Thus, in order to show that the pattern retrieval capacity is exponential in n, all we need to demonstrate is that there exists a training set $\mathcal{X}$ with C patterns of length n for which $C \propto a^{rn}$, for some $a > 1$ and $0 < r$.

Theorem 1 (Karbasi et al., 2013). *Let $\mathcal{X}$ be a $\mathcal{C} \times n$ matrix, formed by $\mathcal{C}$ vectors of length n with entries from the set $\mathcal{Q}$. Furthermore, let $k = rn$ for some $0 < r < 1$. Then there exists a set of vectors for which $\mathcal{C} = a^{rn}$, with $a > 1$, and rank$(\mathcal{X}) = k < n$.*

The proof is constructive: we create a data set $\mathcal{X}$ such that it can be memorized by the proposed neural network and satisfies the required properties; the subpatterns form a subspace, and pattern entries are integer values from the set $\mathcal{Q} = \{0, \ldots, Q-1\}$. The complete proof can be found in Karbasi et al. (2013).

4 Recall Performance Analysis

Now let us analyze recall error performance. The following lemma shows that if φ and ψ are chosen properly, then in the absence of external errors, the constraints remain satisfied and internal noise cannot result in violations. This is a crucial property for algorithm 2, as it allows one to determine whether a cluster has successfully eliminated external errors (step 4 of algorithm) by merely checking the satisfaction of all constraint nodes.

Lemma 1. *In the absence of external errors, the probability that a constraint neuron (resp. pattern neuron) in cluster ℓ makes a wrong decision due to its internal noise is given by $\pi_0^{(\ell)} = \max\left(0, \frac{\nu-\psi}{\nu}\right)$ (resp. $P_0^{(\ell)} = \max\left(0, \frac{\upsilon-\varphi}{\upsilon}\right)$).*

Proof. To calculate the probability that a constraint node makes a mistake when there are no external errors, consider constraint node i whose decision parameter will be

$$h_i^{(\ell)} = \left(W^{(\ell)} \cdot x^{(\ell)}\right)_i + v_i = v_i.$$

Therefore, the probability of making a mistake will be

$$\pi_0^{(\ell)} = \Pr\{|v_i| > \psi\} = \max\left(0, \frac{\nu - \psi}{\nu}\right). \tag{4.1}$$

Thus, to make $\pi_0^{(\ell)} = 0$, we will select $\psi > \nu$. Note that this might not be possible in all cases since, as we will see, the minimum absolute value of network weights should be at least ψ; if ψ is too large, we might not be able to find a proper set of weights. Nevertheless, and assuming that it is possible to choose a proper ψ, we will have

$$\pi^{(0)} = 0. \tag{4.2}$$

Knowing that the constraint will not send any nonzero messages in the absence of external noise, we focus on the pattern neurons in the same circumstance. A given pattern node $x_j^{(\ell)}$ will receive a zero from all its neighbors among the constraint nodes. Therefore, its decision parameter will be $g_j^{(\ell)} = u_j$. As a result, a mistake could happen if $|u_j| \geq \varphi$. The probability of this event is given by

$$P_0^{(\ell)} = \Pr\{|u_i| > \varphi\} = \max\left(0, \frac{\upsilon - \varphi}{\varphi}\right). \tag{4.3}$$

Therefore, to make $P_0^{(\ell)}$ go to zero, we must select $\varphi \geq \upsilon$.

In the sequel, we assume $\varphi > \upsilon$ and $\psi > \nu$ so that $\pi_0^{(\ell)} = 0$ and $P_0^{(\ell)} = 0$. However, an external error combined with internal noise may still push neurons to an incorrect state.

Given the above lemma and our neural architecture, we can prove the following surprising result: in the asymptotic regime of increasing numbers of iterations of algorithm 2, a neural network with internal noise outperforms one without, with the pattern retrieval capacity remaining intact. Let us define the fraction of errors corrected by the noiseless and noisy neural network (parameterized by υ and ν) after T iterations of algorithm 2 by $\Lambda(T)$ and $\Lambda_{\upsilon,\nu}(T)$, respectively. Note that both $\Lambda(T) \leq 1$ and $\Lambda_{\upsilon,\nu}(T) \leq 1$ are nondecreasing sequences of T. Hence, their limiting values are well defined: $\lim_{T\to\infty} \Lambda(T) = \Lambda^*$ and $\lim_{T\to\infty} \Lambda_{\upsilon,\nu}(T) = \Lambda^*_{\upsilon,\nu}$.

Theorem 2. *Let us choose φ and ψ so that $\pi_0^{(\ell)} = 0$ and $P_0^{(\ell)} = 0$ for all $\ell \in \{1, \ldots, L\}$. For the same realization of external errors, we have $\Lambda^*_{\upsilon,\nu} \geq \Lambda^*$.*

Proof. We first show that the noisy network can correct any external error pattern that the noiseless counterpart can correct in the $T \to \infty$ limit. If the noiseless decoder succeeds, then there is a nonzero probability P that the noisy decoder succeeds in a given round as well (corresponding to the case that noise values are rather small). Since we do not introduce new errors during the application of algorithm 2, the number of errors in the new rounds is smaller than or equal to the previous round; hence, the probability of success is lower-bounded by P. If algorithm 2 is applied T times, then the probability of correcting the external errors at the end of round T is $P + P(1-P) + \cdots + P(1-P)^{T-1} = 1 - (1-P)^T$. Since $P > 0$, for $T \to \infty$ this probability tends to 1.

Now we turn our attention to cases where the noiseless network fails in eliminating external errors and show that there exist external error patterns, called stopping sets, for which the noisy network is capable of eliminating them while the noiseless network has failed (see section A.1 for further explication). Assuming that each cluster can eliminate i external errors in their domain and in the absence of internal noise, stopping sets correspond to noise patterns in which each cluster has more than i errors.[4] Then algorithm 2 cannot proceed any further. However, in the noisy network, there is a chance that in one of the rounds, the noise acts favorably and the cluster could correct more than i errors.[5] In this case, if the probability of getting out of the stopping set is P in each round, for some $P > 0$, then a similar argument to the previous case shows that $P \to 1$ when $T \to \infty$.

It should be noted that if the amount of internal noise or external errors is too high, the noisy architecture will eventually get stuck just like the noiseless network would. The high-level idea of why a noisy network outperforms a noiseless one comes from understanding stopping sets—realizations of external errors where the iterative algorithm 2 cannot correct them all. We showed that the stopping set shrinks as we add internal noise, and so the supposedly harmful internal noise helps algorithm 2 to avoid stopping sets. Section A.1 illustrates this notion further.

Theorem 2 suggests that the only possible downside to using a noisy network is its possible running time in eliminating external errors: the noisy neural network may need more iterations to achieve the same error correction performance. Interestingly, our empirical experiments show that in certain scenarios, even the running time improves when using a noisy network.

Theorem 2 indicates that noisy neural networks (under our model) outperform noiseless ones, but does not specify the level of errors that such networks can correct. Now we derive a theoretical upper bound on error correction performance. To this end, let P_{c_i} be the average probability that a cluster can correct i external errors in its domain. The following theorem gives a simple condition under which algorithm 2 can correct a linear fraction of external errors (in terms of n) with high probability. The condition involves $\tilde{\lambda}$ and $\tilde{\rho}$, the degree distributions of the contracted graph $\tilde{G}$.

Theorem 3. *Under the assumptions that graph $\widetilde{G}$ grows large and it is chosen randomly with degree distributions given by $\widetilde{\lambda}$ and $\widetilde{\rho}$, algorithm 2 is successful if*

$$\epsilon\widetilde{\lambda}\left(1-\sum_{i\geq 1}P_{c_i}\frac{z^{i-1}}{i!}\cdot\frac{d^{i-1}\widetilde{\rho}(1-z)}{dz^{i-1}}\right) < z,\ \ for\ z\in[0,\epsilon]. \tag{4.4}$$

Proof. The proof is based on the density evolution technique (Richardson & Urbanke, 2008). Without loss of generality, assume we have P_{c_1}, P_{c_2}, and P_{c_3} (and $P_{c_i} = 0$ for $i > 3$), but the proof can easily be extended if we have P_{c_i}

[4] In Figure 2, we will see that $i = 2$ in this case.

[5] This is reflected in Figure 2, where the value of P_{c_i} is larger when the network is noisy.

for $i > 3$. Let $\Pi(t)$ be the average probability that a super-constraint node sends a failure message: it cannot correct external errors lying in its domain. Then the probability that a noisy pattern neuron with degree d_i sends an erroneous message to a particular neighbor among super-constraint nodes is equal to the probability that none of its other neighboring super-constraint nodes could have corrected its error:

$$P_i(t) = p_e(\Pi(t))^{d_i-1}.$$

Averaging over d_i we find the average probability of error in iteration t:

$$z(t+1) = p_e\widetilde{\lambda}(\Pi(t)). \tag{4.5}$$

Now consider a cluster ℓ that contains d_ℓ pattern neurons. This cluster will not send a failure message over its edge to a noisy pattern neuron in its domain with probability

1. P_{c_1} if it is not connected to any other noisy neuron
2. P_{c_2} if it is connected to exactly one other constraint neuron
3. P_{c_3} if it is connected to exactly two other constraint neurons
4. 0 if it is connected to more than two other constraint neurons

Thus,

$$\Pi^{(\ell)}(t) = 1 - P_{c_1}(1-z(t))^{d_\ell-1} - P_{c_2}\binom{d_\ell-1}{1}z(t)\,(1-z(t))^{d_\ell-2} - P_{c_3}\binom{d_\ell-1}{2}z(t)^2\,(1-z(t))^{d_\ell-3}.$$

Averaging over d_ℓ yields

$$\Pi(t) = \mathbb{E}_{d_\ell}\left(\Pi^{(\ell)}(t)\right) = 1 - P_{c_1}\rho(1-z(t)) - P_{c_2}z\rho'(1-z(t)) - \frac{1}{2}P_{c_2}z(t)^2\rho''(1-z(t)), \tag{4.6}$$

where $\rho'(x)$ and $\rho''(x)$ are derivatives of the function $\rho(x)$ with respect to x.

Equations 4.5 and 4.6 yield the value of $z(t+1)$ as a function of $z(t)$. We calculate the final error probability as $\lim_{t\to\infty} z(t)$; for $\lim_{t\to\infty} z(t) \to 0$, it is sufficient to have $z(t+1) < z(t)$, which proves the theorem.

Theorem 3 holds when the decision subgraphs for the pattern neurons in graph $\widetilde{G}$ are tree-like for a depth of τL, where τ is the total number of number of iterations performed by algorithm 2 (Richardson & Urbanke, 2008).

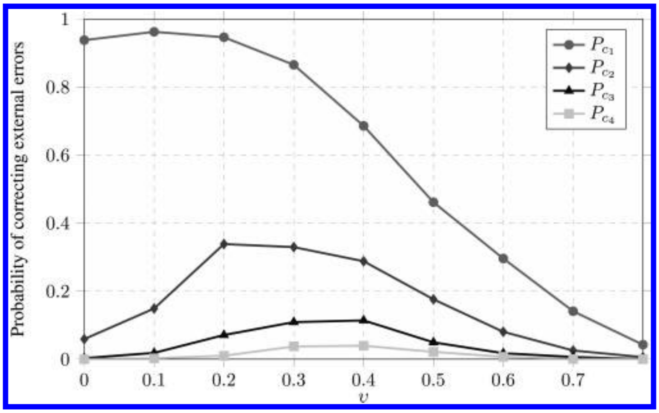


Figure 2: The value of P_{c_i} as a function of pattern neurons noise υ for $i = 1, \dots, 4$. The noise at constraint neurons is assumed to be zero ($\nu = 0$).

Theorem 3 states that for any fraction of errors $\Lambda_{\upsilon,\nu} \leq \Lambda^*_{\upsilon,\nu}$ that satisfies the above recursive formula, algorithm 2 will be successful with probability close to one. Note that the first fixed point of the above recursive equation dictates the maximum fraction of errors $\Lambda^*_{\upsilon,\nu}$ that our model can correct. For the special case of $P_{c_1} = 1$ and $P_{c_i} = 0$, for all $i > 1$, we obtain $\epsilon\widetilde{\lambda}\,(1 - \widetilde{\rho}(1 - z)) < z$, the same condition given in Karbasi et al. (2013). Theorem 4 takes into account the contribution of all P_{c_i} terms, and as we will see, their values change as we incorporate the effect of internal noise υ and ν. Our results show that the maximum value of P_{c_i} does not occur when the internal noise is equal to zero, $\upsilon = \nu = 0$, but instead when the neurons are contaminated with internal noise. As an example, Figure 2 illustrates how P_{c_i} behaves as a function of υ in the network considered (note that maximum values are not at $\upsilon = 0$). This finding suggests that even individual clusters are able to correct more errors in the presence of internal noise.

To estimate the P_{c_i} values, we use numerical approaches.[6] Given a set of clusters $W^{(1)}, \dots, W^{(L)}$, for each cluster, we randomly corrupt i pattern neurons with ±1 noise. Then we apply algorithm 1 over this cluster and calculate the success rate once finished. We take the average of this rate over all clusters to end up with P_{c_i}. The results of this approach are shown in Figure 2, where the value of P_{c_i} is shown for $i = 1, \dots, 4$ and various noise amounts at the pattern neurons (specified by parameter υ).

[6] Section A.2 derives an analytical upper bound to estimate P_{c_1} but this requires approximations that are loose.

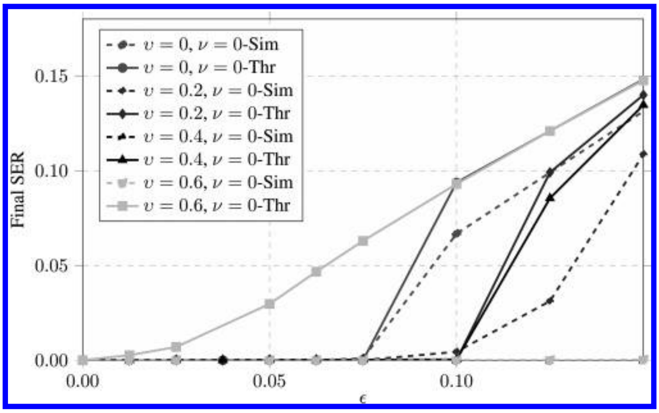


Figure 3: The final SER for a network with $n = 400$, $L = 50$. The $\upsilon = 0$ curves correspond to the noiseless neural network.

4.1 Simulations. Now we consider simulation results for a finite system. To learn the subspace constraints 2.1 for each cluster $G^{(\ell)}$, we use the learning algorithm in Karbasi et al. (2013). Henceforth, we assume that the weight matrix W is known and given. In our setup, we consider a network of size $n = 400$ with $L = 50$ clusters. We have 40 pattern nodes and 20 constraint nodes in each cluster on average. External error is modeled by randomly generated vectors z with entries ± 1 with probability ϵ and 0 otherwise. Vector z is added to the correct patterns, which satisfy equation 2.1. For recall, algorithm 2 is used, and results are reported in terms of symbol error rate (SER) as the level of external error (ϵ) or internal noise (υ, ν) is changed; this involves counting positions where the output of algorithm 2 differs from the correct pattern.

4.1.1 Symbol Error Rate as a Function of Internal Noise. Figure 3 illustrates the final SER of our algorithm for different values of υ and ν. Remember that υ and ν quantify the level of noise in pattern and constraint neurons, respectively. Dashed lines in Figure 3 are simulation results, whereas solid lines are theoretical upper bounds provided in this letter. There is a threshold phenomenon such that SER is negligible for $\epsilon \leq \epsilon^*$ and grows beyond this threshold. As expected, simulation results are better than the theoretical bounds. In particular, the gap is relatively large as υ moves toward one.

A more interesting trend in Figure 3 is the fact that internal noise helps in achieving better performance, as predicted by theoretical analysis (see theorem 2). Notice how ϵ^* moves toward one as ν increases.

This phenomenon is inspected more closely in Figure 4, where ϵ is fixed to 0.125 while υ and ν vary. Figures 5a and 5b display projected versions of the surface plot to investigate the effect of υ and ν separately. As we see

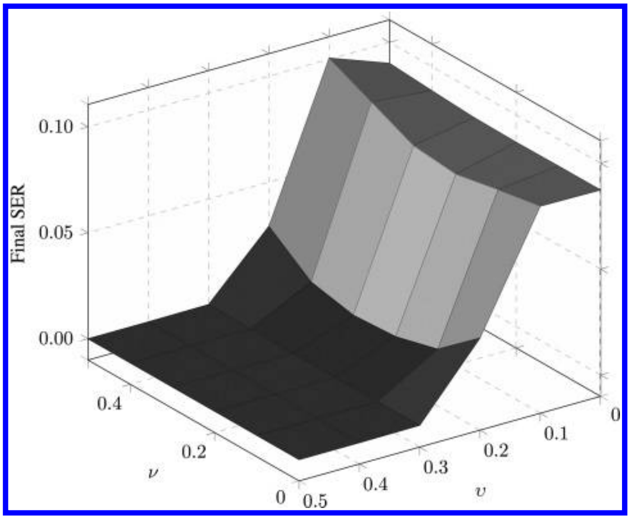
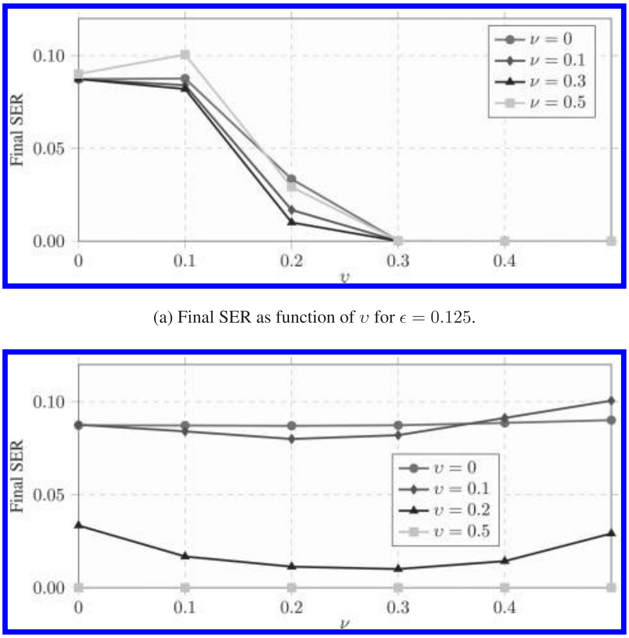


Figure 4: The final symbol error probability when $\epsilon = 0.125$ as a function of internal noise parameters at the pattern and constraint neurons, denoted by υ and ν, respectively.



(a) Final SER as function of υ for $\epsilon = 0.125$.

(b) The effect of ν on the final SER for $\epsilon = 0.125$

Figure 5: The final symbol error probability as a function of internal noise parameters at pattern and constraint neurons for $\epsilon = 0.125$.

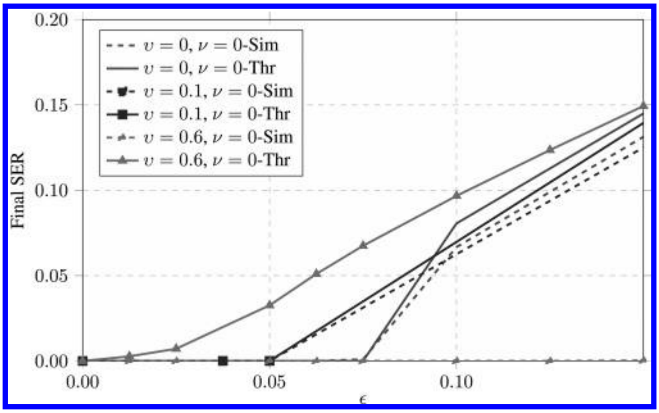


Figure 6: The final SER for a network with $n = 400$, $L = 50$ and noise values chosen from $\{-3, -2, \ldots, 2, 3\}$. The $\nu = 0$ curves correspond to the noiseless neural network.

again, a moderate amount of internal noise at both pattern and constraint neurons improves performance. There is an optimum point (υ^*, ν^*) for which the SER reaches its minimum. Figure 5b indicates, for instance, that $\nu^* \approx 0.25$, beyond which SER deteriorates. There is greater sensitivity to noise υ in the pattern neurons, reminiscent of results for decoding circuits with internal noise (Tabatabaei Yazdi, Cho, & Dolecek, 2013).

4.1.2 Larger Noise Values. So far, we have investigated the performance of the recall algorithm when noise values are limited to ±1. Although this choice facilitates the analysis of the algorithm and increases error correction speed, our analysis is valid for larger noise values. Figure 6 illustrates the SER for the same scenario as before but with noise values chosen from $\{-3, -2, \ldots, 2, 3\}$. We see exactly the same behavior as we witnessed for ±1 noise values.

4.2 Recall Time as a Function of Internal Noise. Figure 7 illustrates the number of iterations performed by algorithm 2 for correcting the external errors when ϵ is fixed to 0.075. We stop whenever the algorithm corrects all external errors or declare a recall error if all errors were not corrected in 40 iterations. Thus, the corresponding areas in the figure where the number of iterations reaches 40 indicates decoding failure. Figures 8a and 8b, projected versions of Figure 7, show the average number of iterations as a function of υ and ν, respectively.

The amount of internal noise drastically affects the speed of algorithm 2. First, from Figures 7 and 8b, observe that running time is more sensitive to noise at constraint neurons than pattern neurons and that the algorithms

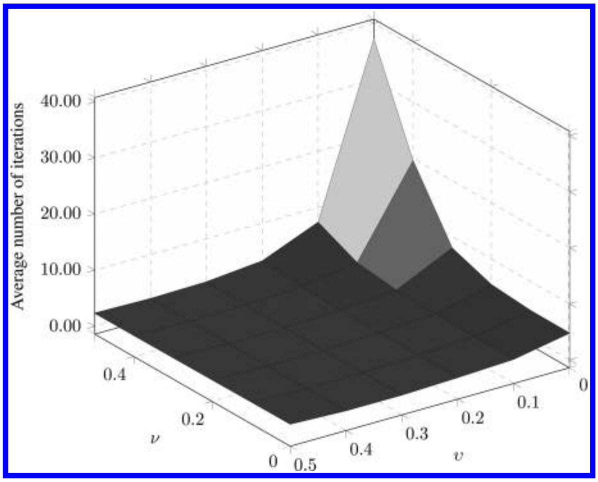
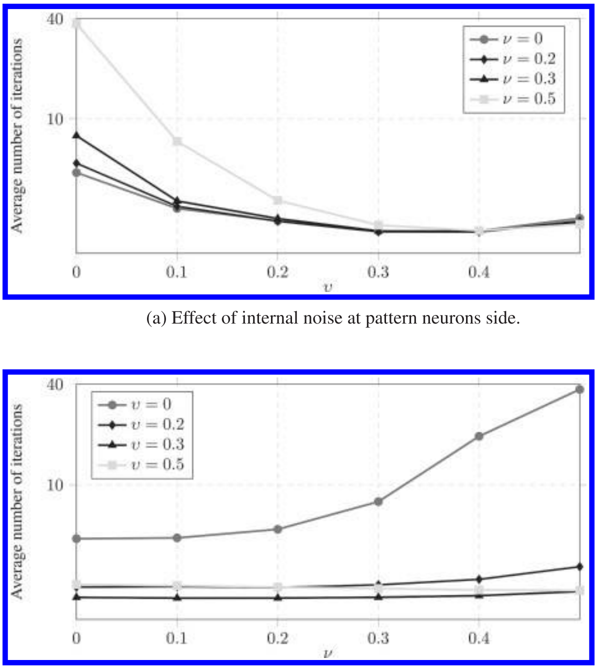


Figure 7: The effect of internal noise on the number of iterations performed by algorithm 2, for different values of υ and ν with $\epsilon = 0.075$.



(a) Effect of internal noise at pattern neurons side.

(b) Effect of internal noise at constraint neurons side.

Figure 8: The effect of internal noise on the number of iterations performed by algorithm 2 for different values of υ and ν with $\epsilon = 0.075$. The average iteration number of 40 indicates the failure of algorithm 2.

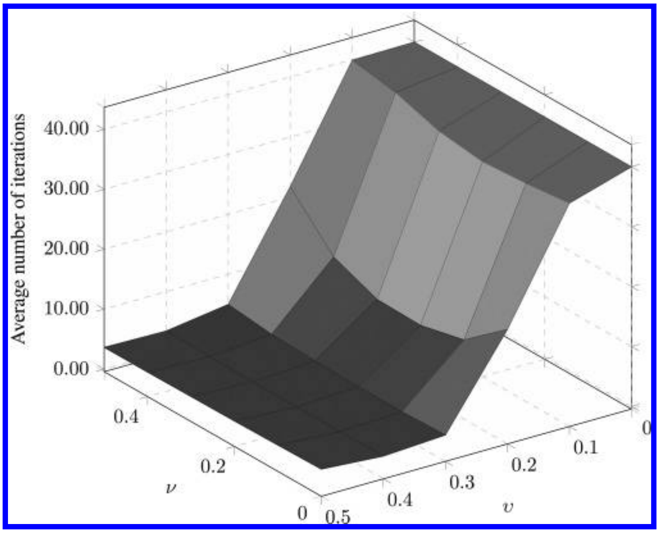
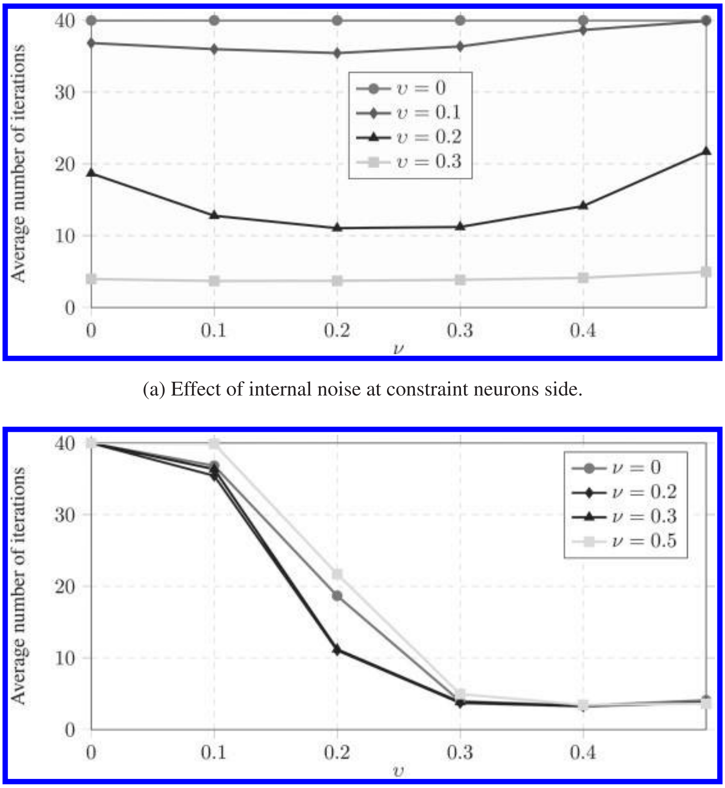


Figure 9: The effect of internal noise on the number of iterations performed by algorithm 2 for different values of υ and ν with $\epsilon = 0.125$.

become slower as noise at constraint neurons is increased. In contrast, note that internal noise at the pattern neurons may improve the running time, as seen in Figure 8a. Ordering of sensitivity to noise in pattern neurons and in constraint neurons is opposite for running time as compared to error probability.

Note that the results presented so far are for the case where the noiseless decoder succeeds as well and its average number of iterations is pretty close to the optimal value (see Figure 7). Figure 9 illustrates the number of iterations performed by algorithm 2 for correcting the external errors when ϵ is fixed to 0.125. In this case, the noiseless decoder encounters stopping sets, while the noisy decoder is still capable of correcting external errors. Here we see that the optimal running time occurs when the neurons have a fair amount of internal noise. Figures 10b and 10a are projected versions of Figure 9 and show the average number of iterations as a function of υ and ν, respectively.

4.3 Effect of Internal Noise on the Performance in the Absence of External Noise. Now we provide results of a study for a slightly modified setting where there is only internal noise and no external errors and further $\varphi < \upsilon$. Thus, the internal noise can now cause neurons to make wrong decisions, even in the absence of external errors. With abuse of notation, we assume pattern neurons are corrupted with a ± 1 noise added to them with probability υ. The rest of the model setting is the same as before.



(a) Effect of internal noise at constraint neurons side.

(b) Effect of internal noise at pattern neurons side.

Figure 10: The effect of internal noise on the number of iterations performed by algorithm 2 for different values of υ and ν with $\epsilon = 0.125$. The average iteration number of 40 indicates the failure of algorithm 2.

Figure 11 illustrates the effect of internal noise as a function of υ and ν, the noise parameters at the pattern and constraint nodes, respectively. This behavior is shown in Figures 12a and 12b for better inspection. Here, we witness the more familiar phenomenon where increasing the amount of internal noise results in a worse performance. This finding emphasizes the importance of choosing update threshold φ and ψ properly, according to lemma 1. See section A.3 for details on choosing thresholds.

5 Discussion

We have demonstrated that associative memories with exponential pattern capacity still work reliably even when built from unreliable hardware,

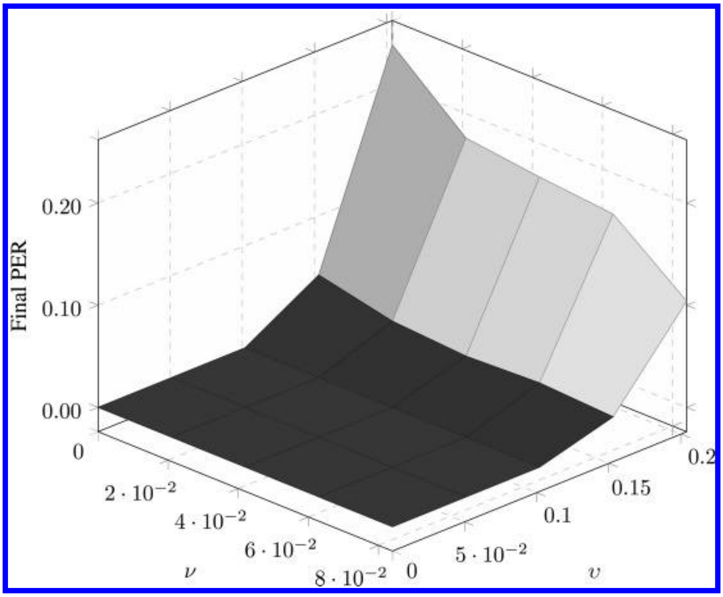
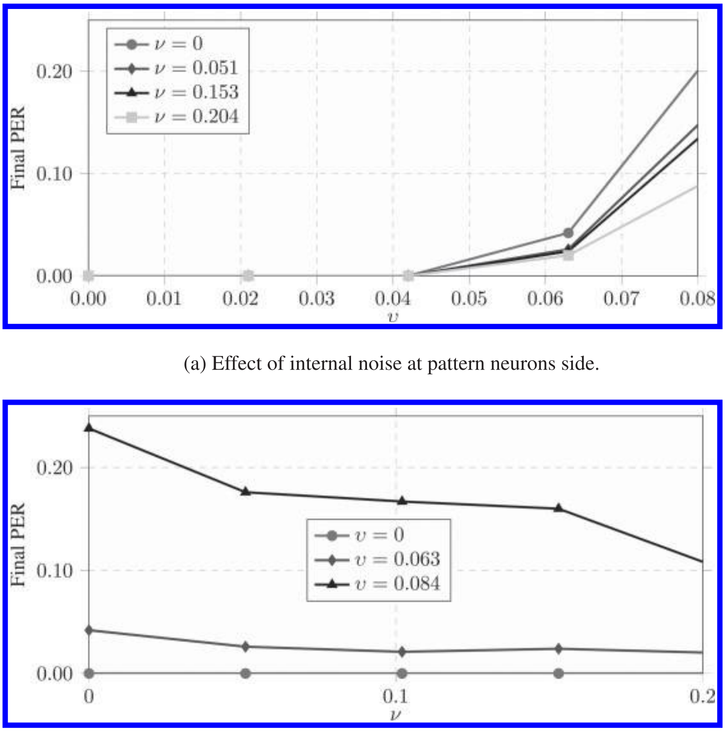


Figure 11: The effect of the internal noise on final pattern error rate (PER) as a function of υ and ν in the absence of external noise.

addressing a major problem in fault-tolerant computing and further arguing for the viability of associative memory models for the (noisy) mammalian brain. After all, brain regions modeled as associative memories, such as the hippocampus and the olfactory cortex, do display internal noise (Koch, 1999; Yoshida, Hayashi, Tateno, & Ishizuka, 2002; McDonnell & Ward, 2011).

We used a particular neuronal computation model. The linear-nonlinear computations of algorithm 1 are nearly identical to message-passing algorithms such as belief propagation and are certainly biologically plausible (Beck & Pouget, 2007; Dayan, Hinton, Neal, & Zemel, 1995; Deneve, 2008; Doya, Ishii, Pouget, & Rao, 2007; Hinton & Sejnowski, 1986; Ma, Beck, Latham, & Pouget, 2006; Litvak & Ullman, 2009). The state reversion computation of algorithm 2 requires keeping a state variable for a short amount of time, which has been suggested as realistic for biological neurons (Druckmann & Chklovskii, 2012), but the general biological plausibility of algorithm 2 remains an open question.

We found a threshold phenomenon for reliable operation, which manifests the tradeoff between the amount of internal noise and the amount of external noise that the system can handle. In fact, we showed that bounded internal noise actually improves the performance of the network in dealing with external errors, up to some optimal value. This is a manifestation of the stochastic facilitation (McDonnell & Ward, 2011) or *noise enhancement*



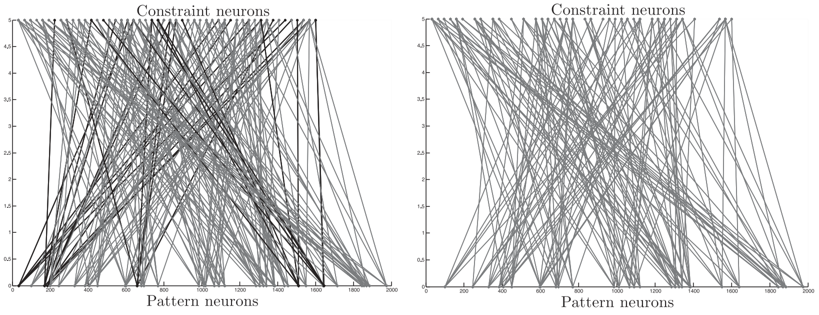
(a) Effect of internal noise at pattern neurons side.

(b) Effect of internal noise at constraint neurons side.

Figure 12: The effect of the internal noise on final pattern error rate (PER) as a function of υ and ν in absence of external noise.

(Chen et al., 2007) phenomenon that has been observed in other neuronal and signal processing systems, providing a functional benefit to variability in the operation of neural systems.

The associative memory design developed here uses thresholding operations in the message-passing algorithm for recall. As part of our investigation, we optimized these neural firing thresholds based on the statistics of the internal noise. As Sarpeshkar (1998) noted in describing the properties of analog and digital computing circuits, "In a cascade of analog stages, noise starts to accumulate. Thus, complex systems with many stages are difficult to build. [In digital systems] round-off error does not accumulate significantly for many computations. Thus, complex systems with many stages are easy to build." One key to our result is capturing this benefit of digital processing (thresholding to prevent the buildup of errors due to internal noise) as well as a modular architecture, which allows us to correct a linear number of external errors (in terms of the pattern length).



(a) $t = 0$ (b) $t \to \infty$

Figure 13: An external noise pattern that contains a stopping set in a noiseless neural circuit. (Left) The original pattern. (Right) The result of the decoding algorithm after sufficient number of iterations where the algorithm gets stuck. Dark-colored pattern nodes are those that are connected to at least one cluster with a single external error. Obviously the stopping set on the right does not have any dark-colored nodes.

This letter has focused on recall; however, learning is the other critical stage of associative memory operation. Indeed, information storage in nervous systems is said to be subject to storage (or learning) noise, in situ noise, and retrieval (or recall) noise (Varshney et al., 2006, Figure 1). It should be noted, however, that there is no essential loss by combining learning noise and in situ noise into what we have called external error herein (cf. Varshney, 2011, note 1 and proposition 1). Thus our basic qualitative result extends to the setting where the learning and stored phases are also performed with noisy hardware.

Going forward, it is of interest to investigate other neural information processing models that explicitly incorporate internal noise and see whether they provide insight into observed empirical phenomena. As an example, we might be able to explain the threshold phenomenon observed in the symbol error rate of human telegraph operators under heat stress (Mackworth, 1946, Figure 2) by invoking a thermal internal noise explanation. Returning to engineering, internal noise in decoders for limited-length error-correcting codes may improve performance as observed herein, since stopping sets are a limiting phenomenon in that setting also.

Appendix

A.1 Illustrating Proof of Theorem 2. Figure 13a illustrates an example of a stopping set over the graph $\widetilde{G}$ in our empirical studies. In the figure, only the nodes corrupted with external noise are shown for clarity. Pattern neurons that are connected to at least one cluster with a single error are dark-colored, and other pattern neurons are light-colored. Figure 13b illustrates the same network but after a number of decoding iterations that result in the algorithm getting stuck. We have a stopping set in which no cluster has a single error and the algorithm cannot proceed further since $P_{c_i} \simeq 0$ for $i > 1$ in a noiseless architecture. Thus, the external error cannot get corrected.

As evident from Figure 13, the stopping set is the result of clusters not being able to correct more than one external error; this is where internal noise might come to the rescue. Interestingly, an "unreliable" neural circuit in which $\upsilon = 0.6$ could easily get out of the stopping set shown in Figure 13b and correct all of the external errors. We try several times to correct errors in a cluster (and overall in the network) while making sure that the algorithm does not introduce new errors itself. Thus, the noise might act in our favor in one of these attempts and the algorithm might be able to avoid the stopping set, as depicted in Figure 13.

A.2 Estimating P_{c_1} Theoretically. To bound P_{c_1}, consider four event probabilities for a cluster:

- $\pi_0^{(\ell)}$ (resp. $P_0^{(\ell)}$): The probability that a constraint neuron (resp. pattern neuron) in cluster ℓ makes a wrong decision due to its internal noise when there is no external noise introduced to cluster ℓ, that is, $\|z^{(\ell)}\|_0 = 0$.
- $\pi_1^{(\ell)}$ (resp. $P_1^{(\ell)}$): The probability that a constraint neuron (resp. pattern neuron) in cluster ℓ makes a wrong decision due to its internal noise when one input error (external noise) is introduced, that is, $\|z^{(\ell)}\|_0 = \|z^{(\ell)}\|_1 = 1$.

Notice $P_{c_1}^{(\ell)} = 1 - P_1^{(\ell)}$.

We derive an upper bound on the probability a constraint node makes a mistake in the presence of one external error.

Lemma 2. *In the presence of a single external error, the probability that a constraint neuron makes a wrong decision due to its internal noise is given by*

$$\pi_1^{(\ell)} \leq \max\left(0, \frac{\nu - (\eta - \psi)}{2\nu}\right),$$

where $\eta = \min_{i,j,W_{ij}\neq 0}(|W_{ij}|)$ *is the minimum absolute value of the nonzero weights in the neural graph and is chosen such that* $\eta \geq \psi$.[7]

[7] This condition can be enforced during simulations as long as ψ is not too large, which itself is determined by the level of constraint neuron internal noise, ν, as we must have $\psi \geq \nu$.

Proof. Without loss of generality, assume it is the first pattern node, $x_1^{(\ell)}$, that is corrupted with noise +1. Now calculate the probability that a constraint node makes a mistake in such circumstances. We need only analyze constraint neurons connected to $x_1^{(\ell)}$ since the situation for other constraint neurons is as when there is no external error. For a constraint neuron j connected to $x_1^{(\ell)}$, the decision parameter is

$$\begin{aligned} h_j^{(\ell)} &= (W^{(\ell)}.(x^{(\ell)} + z^{(\ell)}))_j + v_j \\ &= 0 + (W^{(\ell)}.z^{(\ell)})_j + v_j \\ &= w_{j1} + v_j. \end{aligned}$$

We consider two error events:

- A constraint node j makes a mistake and does not send a message at all. The probability of this event is denoted by $\pi_1^{(1)}$.
- A constraint node j makes a mistake and sends a message with the opposite sign. The probability of this event is denoted by $\pi_2^{(1)}$.

We first calculate the probability of $\pi_2^{(1)}$. Without loss of generality, assume the $w_{j1} > 0$ so that the probability of an error of type 2 is as follows (the case for $w_{j1} < 0$ is exactly the same):

$$\begin{aligned} \pi_2^{(1)} &= \Pr\{w_{ji} + v_j < -\psi\} \\ &= \max\left(0, \frac{\nu - (\psi + w_{j1})}{2\nu}\right). \end{aligned} \tag{A.1}$$

However, since $\psi > \nu$ and $w_{j1} > 0$, then $\nu - (\psi + w_{j1}) < 0$ and $\pi_2^{(1)} = 0$. Therefore, the constraint neurons will never send a message that has an opposite sign to what it should have. All that remains is to calculate the probability they remain silent by mistake.

To this end, we have

$$\begin{aligned} \pi_1^{(1)} &= \Pr\{|w_{ji} + v_j| < \psi\} \\ &= \max\left(0, \frac{\nu + \min(\psi - w_{j1}, \nu)}{2\nu}\right). \end{aligned} \tag{A.2}$$

This can be simplified if we assume that the absolute values of all weights in the network are bigger than a constant $\eta > \psi$. Then equation A.2 will simplify to

$$\pi_1^{(1)} \leq \max\left(0, \frac{\nu - (\eta - \psi)}{2\nu}\right). \tag{A.3}$$

Putting equations A.2 and A.3 together, we obtain

$$\pi^{(1)} \leq \max\left(0, \frac{\nu - (\eta - \psi)}{2\nu}\right). \tag{A.4}$$

In the case $\eta - \psi > \nu$, we could even manage to make this probability equal to zero. However, we will leave it as is and use equation A.4 to calculate $P_1^{(\ell)}$.

A.2.1 Calculating $P_1^{(\ell)}$. We start by calculating the probability that a noncorrupted pattern node $x_j^{(\ell)}$ makes a mistake, which is to change its state in round 1. Let us denote this probability by $q_1^{(\ell)}$. To calculate $q_1^{(\ell)}$, assume $x_j^{(\ell)}$ has degree d_j and it has b common neighbors with $x_1^{(\ell)}$, the corrupted pattern node.

Out of these b common neighbors, b_c will send ±1 messages and the others will, mistakenly, send nothing. Thus, the decision-making parameter of pattern node j, $g_j^{(\ell)}$, will be bounded by

$$g_j^{(\ell)} = \frac{(\text{sign}(W^{(\ell)})^\top \cdot y^{(\ell)})_j}{d_j} + u_j. \leq \frac{b_c}{d_j} + u_j.$$

We denote $\left(\text{sign}(W^{(\ell)})^\top \cdot y^{(\ell)}\right)_j$ by o_j for brevity from this point on.

In this circumstance, a mistake happens when $|g_j^{(\ell)}| \geq \varphi$. Thus

$$\begin{aligned} q_1^{(\ell)} &= \Pr\{|g_j^{(\ell)}| \geq \varphi | \deg(a_j) = d_j \& |\mathcal{N}(x_1) \cap \mathcal{N}(a_j)| = a\} \\ &= \Pr\left\{\frac{o_j}{d_j} + u_j \geq \varphi\right\} + \Pr\left\{\frac{o_j}{d_j} + u_j \leq -\varphi\right\}, \end{aligned} \tag{A.5}$$

where $\mathcal{N}(a_i)$ represents the neighborhood of pattern node a_i among constraint nodes.

By simplifying equation A.5, we get

$$q_1^{(\ell)}(o_j) = \begin{cases} +1, & \text{if } |o_j| \geq (\upsilon + \varphi)d_j \\ \max\left(0, \dfrac{\upsilon - \varphi}{\upsilon}\right), & \text{if } |o_j| \leq |\upsilon - \varphi| d_j. \\ \dfrac{\upsilon - (\varphi - o_j/d_j)}{2\upsilon}, & \text{if } |o_j - \varphi d_j| \leq \upsilon d_j \\ \dfrac{\upsilon - (\varphi + o_j/d_j)}{2\upsilon}, & \text{if } |o_j + \varphi d_j| \leq \upsilon d_j \end{cases}$$

We now average this equation over o_j, b_c, b, and d_j. To start, suppose that out of the b_c nonzero messages node a_j receives, e of them have the same sign as the link they are being transmitted over. Thus, we will have $o_j = e - (b_c - e) = 2e - b_c$. Assuming the probability of having the same sign for each message is 1/2, the probability of having e equal signs out of b_c elements will be $\binom{b_c}{e}\left(\frac{1}{2}\right)^{b_c}$. Thus, we will get

$$\bar{q}_1^{(\ell)} = \sum_{e=0}^{b_c} \binom{b_c}{e} \left(\frac{1}{2}\right)^{b_c} q_1^{(\ell)}(2e - b_c). \tag{A.6}$$

Now note that the probability of having $a - b_c$ mistakes from the constraint side is given by $\binom{b}{b_c}(\pi_1^{(\ell)})^{b-b_c}(1-\pi_1^{(\ell)})^{b_c}$. With some abuse of notation, we get

$$\bar{q}_1^{(\ell)} = \sum_{b_c=0}^{b} \binom{b}{b_c} (\pi_1^{(\ell)})^{b-b_c}(1-\pi_1^{(\ell)})^{b_c} \sum_{e=0}^{b_c} \binom{b_c}{e} \left(\frac{1}{2}\right)^{b_c} q_1^{(\ell)}(2e - b_c). \tag{A.7}$$

Finally, the probability that a_j and x_1 have b common neighbors can be approximated by $\binom{d_j}{b}(1 - \bar{d}^{(\ell)}/m_\ell)^{d_j-b}(\bar{d}^{(\ell)}/m_\ell)^b$, where $\bar{d}^{(\ell)}$ is the average degree of pattern nodes. Thus (again abusing some notation), we obtain

$$\bar{q}_1^{(\ell)} = \sum_{b=0}^{d_j} p_b \sum_{b_c=0}^{b} p_{b_c} \sum_{e=0}^{b_c} \binom{b_c}{e} \left(\frac{1}{2}\right)^{b_c} q_1^{(\ell)}(2e - b_c), \tag{A.8}$$

where $q_1^{(\ell)}(2e - b_c)$ is given by equation A.5, p_b is the probability of having b common neighbors and is estimated by $\binom{d_j}{b}(1 - \bar{d}^{(\ell)}/m_\ell)^{d_j-b}(\bar{d}^{(\ell)}/m_\ell)^b$, with $\bar{d}^{(\ell)}$ being the average degree of pattern nodes in cluster ℓ. Furthermore, p_{b_c} is the probability of having $b - b_c$ out of these b nodes making mistakes. Hence, $p_{b_c} = \binom{b}{b_c}(\pi_1^{(\ell)})^{b-b_c}(1-\pi_1^{(\ell)})^{b_c}$. We will not simplify equation A.8 any further and use it as it is in our numerical analysis in order to obtain the best parameter φ.

Now we turn our attention to the probability that the corrupted node, x_1, makes a mistake: either not to update at all or to update itself in the wrong direction. Recalling that we have assumed the external noise term in x_1 to be a $+1$ noise, the wrong direction would be for node x_1 to increase its current value instead of decreasing it. Furthermore, we assume that out of d_1 neighbors of x_1, some j of them have made a mistake and will not send any messages to x_1. Thus, the decision parameter of x_1 will be

$g_1^{(\ell)} = u + (d_1 - j)/d_1$. Denoting the probability of making a mistake at x_1 by $q_2^{(\ell)}$, we get

$$\begin{aligned} q_2^{(\ell)} &= \Pr\{g_1^{(\ell)} \leq \varphi | \deg(x_1) = d_1 \text{ and } j \text{ errors in constraints}\} \\ &= \Pr\left\{\frac{d_1 - j}{d_1} + u < \varphi\right\}, \end{aligned} \tag{A.9}$$

which simplifies to

$$q_2^{(\ell)}(j) = \begin{cases} +1, & \text{if } |j| \geq (1 + \upsilon - \varphi)d_1 \\ \max\left(0, \dfrac{\upsilon - \varphi}{\upsilon}\right), & \text{if } |j| \leq (1 - \upsilon - \varphi)d_1. \\ \dfrac{\upsilon + \varphi - (d_1 - j)/d_1}{2\upsilon}, & \text{if } |\varphi d_1 - (d_1 - j)| \leq \upsilon d_1 \end{cases} \tag{A.10}$$

Noting that the probability of making j mistakes on the constraint side is $\binom{d_1}{j}(\pi_1^{(\ell)})^j(1 - \pi_1^{(\ell)})^{d_1 - j}$, we get

$$\bar{q}_2^{(\ell)} = \sum_{j=0}^{d_1} \binom{d_1}{j} \left(\pi_1^{(\ell)}\right)^j \left(1 - \pi_1^{(\ell)}\right)^{d_1 - j} q_2^{(\ell)}(j), \tag{A.11}$$

where $q_2^{(\ell)}(j)$ is given by equation A.11.

Putting the above results together, the overall probability of making a mistake on the side of pattern neurons when we have one bit of external noise is

$$P_1^{(\ell)} = \frac{1}{n^{(\ell)}} \bar{q}_2^{(\ell)} + \frac{n^{(\ell)} - 1}{n^{(\ell)}} \bar{q}_1^{(\ell)}. \tag{A.12}$$

Finally, the probability that cluster ℓ could correct one error is that all neurons take the correct decision,

$$P_{c_1}^{(\ell)} = \left(1 - P_1^{(\ell)}\right)^{n^{(\ell)}},$$

and the average probability that clusters could correct one error is simply

$$P_{c_1} = \mathbb{E}_\ell\left(P_{c_1}^{(\ell)}\right). \tag{A.13}$$

We use this equation in order to find the best update threshold φ.

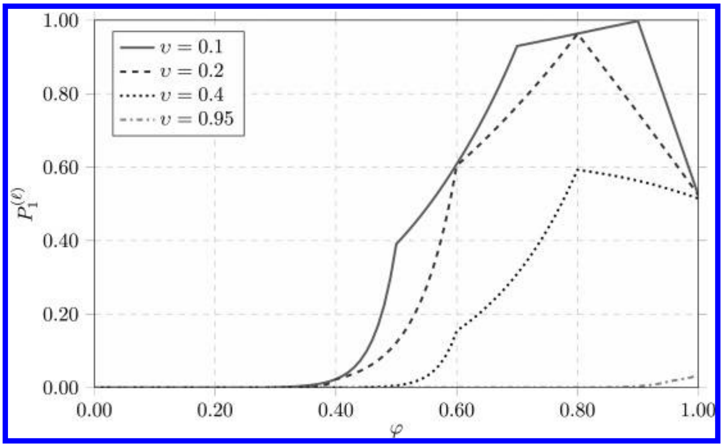


Figure 14: The behavior of P_{c_1} as a function of φ for different values of noise parameter, υ. Here, $\pi_{(1)} = 0.01$.

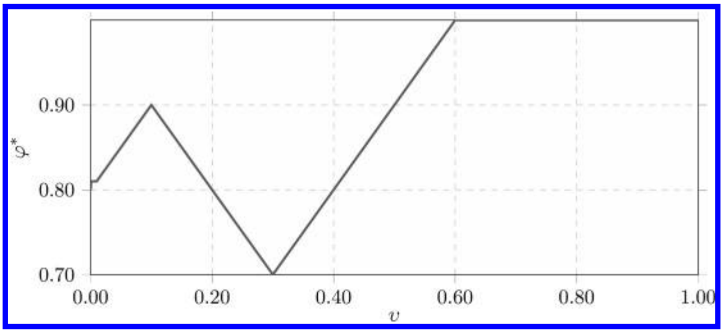


Figure 15: The behavior of φ^* as a function of υ for $\pi_1 = 0.01$.

A.3 Choosing Proper φ. We now apply numerical methods to equation A.12 to find the best φ for different values of noise parameter υ. Figures 14 through 16 show the best choice for the parameter φ. The update threshold on the constraint side is chosen such that $\psi > \nu$. In each figure, we have illustrated the final probability of making a mistake, $P_1^{(\ell)}$, for comparison.

Figure 14 illustrates the behavior of the average probability of correcting a single error, P_{c_1}, as a function of φ for different values of υ and for $\pi_1 = 0.01$. The interesting trend here is that in all cases, φ^*, the update threshold that gives the best result, is chosen such that it is quite large. This is in line with our expectation because a small φ will result in noncorrupted nodes updating their states more frequently. On the other hand, a very large φ will prevent the corrupted nodes from correcting their states, especially if there some mistakes are made on the constraint side, $\pi_1^{(\ell)} > 0$. Therefore, since

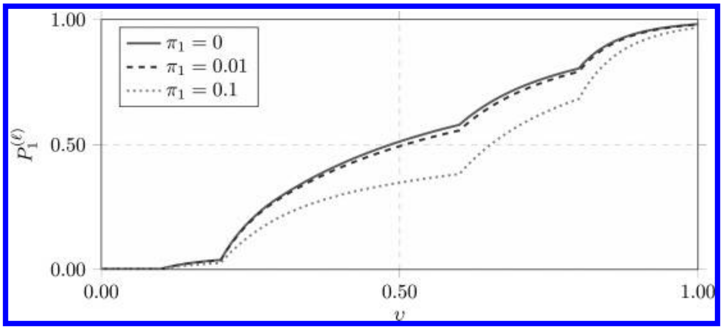


Figure 16: The optimum P_{e_1} as a function of υ for different values π_1.

we have many more noncorrupted nodes than corrupted nodes, it is best to choose a rather high φ but not too high. Also note that when $\pi_1^{(\ell)}$ is very high, there are no values of υ for which error-free storage is possible.

Figure 15 illustrates the exact behavior of φ^* against υ for the case where $\phi_1 = 0$. As can be seen from the figure, φ should be quite large. Figure 16 illustrates $P_{e_1} = 1 - P_{c_1}$ for the best chosen threshold, φ^*, as a function of υ for various choices of π_1.

Acknowledgments

We thank S. S. Venkatesh for telling us about Biswas (1993). A.K. thanks ETH for the fellowship grant.

References

Amit, D. J. (1992). *Modeling brain function*. Cambridge: Cambridge University Press.

Amit, D. J., & Fusi, S. (1994). Learning in neural networks with material synapses. *Neural Comput.*, *6*(5), 957–982.

Baram, Y. (1991). Encoding unique global minima in nested neural networks. *IEEE Trans. Inf. Theory*, *37*(4), 1158–1162.

Beck, J. M., & Pouget, A. (2007). Exact inferences in a neural implementation of a hidden Markov model. *Neural Comput.*, *19*(5), 1344–1361.

Biswas, S. (1993). *A performance analysis of sparse neural associative memory*. Doctoral dissertation, University of Pennsylvania.

Brunel, N., Hakim, V., Isope, P., Nadal, J.-P., & Barbour, B. (2004). Optimal information storage and the distribution of synaptic weights: Perceptron versus Purkinje cell. *Neuron*, *43*(5), 745–757.

Candés, E. J., & Tao, T. (2006). Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE Trans. Inf. Theory*, *52*(12), 5406–5425.

Chen, H., Varshney, P. K., Kay, S. M., & Michels, J. H. (2007). Theory of the stochastic resonance effect in signal detection: Part I—fixed detectors. *IEEE Trans. Signal Process.*, *55*(7), 3172–3184.

Dayan, P., Hinton, G. E., Neal, R. M., & Zemel, R. S. (1995). The Helmholtz machine. *Neural Comput.*, *7*(5), 889–904.

Deneve, S. (2008). Bayesian spiking neurons I: Inference. *Neural Comput.*, *20*(1), 91–117.

Destexhe, A., & Rudolph-Lilith, M. (2012). *Neuronal noise*. New York: Springer.

Doya, K., Ishii, S., Pouget, A., & Rao, R. P. N. (2007). *Bayesian brain: Probabilistic approaches to neural coding*. Cambridge, MA: MIT Press.

Druckmann, S., & Chklovskii, D. B. (2012). Neuronal circuits underlying persistent representations despite time varying activity. *Curr. Biol.*, *22*(22), 2095–2103.

Faisal, A. A., Selen, L. P. J., & Wolpert, D. M. (2008). Noise in the nervous system. *Nat. Rev. Neurosci.*, *9*(4), 292–303.

Fletcher, A. K., Rangan, S., Varshney, L. R., & Bhargava, A. (2011). Neural reconstruction with approximate message passing (NeuRAMP). In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, & K. Weinberger (Eds.), *Advances in neural information processing systems*, *24* (pp. 2555–2563). Cambridge, MA: MIT Press.

Gripon, V., & Berrou, C. (2011). Sparse neural networks with large learning diversity. *IEEE Trans. Neural Netw.*, *22*(7), 1087–1096.

Hebb, D. O. (1949). *The organization of behavior: A neuropsychological theory*. New York: Wiley.

Hinton, G. E., & Sejnowski, T. J. (1986). Learning and relearning in Boltzmann machines. In D. E. Rumelhart & J. L. McLelland (Eds.), *Parallel distributed processing: Explorations in the microfoundations of cognition, Vol. 1: Foundations* (pp. 282–317). Cambridge, MA: MIT Press.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proc. Natl. Acad. Sci. U.S.A.*, *79*(8), 2554–2558.

Jankowski, S., Lozowski, A., & Zurada, J. M. (1996). Complex-valued multistate neural associative memory. *IEEE Trans. Neural Netw.*, *7*(6), 1491–1496.

Karbasi, A., Salavati, A. H., & Shokrollahi, A. (2013). Iterative learning and denoising in convolutional neural associative memories. In *Proc. 30th Int. Conf. Mach. Learn. (ICML 2013)* (pp. 445–453). Madison, WI: Omnipress.

Koch, C. (1999) *Biophysics of computation: Information processing in single neurons*. New York: Oxford University Press.

Koulakov, A. A., & Rinberg, D. (2011). Sparse incomplete representations: A potential role of olfactory granule cells. *Neuron*, *72*(1), 124–136.

Kuznetsov, A. V. (1973). Information storage in a memory assembled from unreliable components. *Probl. Inf. Transm.*, *9*(3), 100–114.

Le, Q. V., Ngiam, J., Chen, Z., Chia, D., Koh, P. W., & Ng, A. Y. (2010). Tiled convolutional neural networks. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, & A. Culotta, (Eds.), *Advances in neural information processing systems*, 23 (pp. 1279–1287). Cambridge, MA: MIT Press.

Lee, D.-L. (2006). Improvements of complex-valued Hopfield associative memory by using generalized projection rules. *IEEE Trans. Neural Netw.*, *17*(5), 1341–1347.

Liljenström, H., & Wu, X.-B. (1995). Noise-enhanced performance in a cortical associative memory model. *Int. J. Neur. Syst.*, *6*(1), 19–29.

Litvak, S., & Ullman, S. (2009). Cortical circuitry implementing graphical models. *Neural Comput.*, *21*(11), 3010–3056.

Luby, M. G., Mitzenmacher, M., Shokrollahi, M. A., & Spielman, D. A. (2001). Efficient erasure correcting codes. *IEEE Trans. Inf. Theory*, *47*(2), 569–584.

Ma, W. J., Beck, J. M., Latham, P. E., & Pouget, A. (2006). Bayesian inference with probabilistic population codes. *Nat. Neurosci.*, *9*(11), 1432–1438.

Mackworth, N. H. (1946). Effects of heat on wireless telegraphy operators hearing and recording Morse messages. *Br. J. Ind. Med.*, *3*(3), 143–158.

McDonnell, M. D., & Ward, L. M. (2011). The benefits of noise in neural systems: Bridging theory and experiment. *Nat. Rev. Neurosci.*, *12*(7), 415–426.

McEliece, R. J., Posner, E. C., Rodemich, E. R., & Venkatesh, S. S. (1987). The capacity of the Hopfield associative memory. *IEEE Trans. Inf. Theory*, *IT-33*(4), 461–482.

Müezzinoğlu, M. K., Güzeliş, C., & Zurada, J. M. (2003). A new design method for the complex-valued multistate Hopfield associative memory. *IEEE Trans. Neural Netw.*, *14*(4), 891–899.

Olshausen, B. A., & Field, D. J. (2004). Sparse coding of sensory inputs. *Curr. Opin. Neurobiol.*, *14*(4), 481–487.

Palm, G. (1980). On associative memory. *Biol. Cybern.*, *36*(1), 19–31.

Peretto, P., & Niez, J. J. (1986). Long term memory storage capacity of multiconnected neural networks. *Biol. Cybern.*, *54*(1), 53–63.

Richardson, T., & Urbanke, R. (2008). *Modern coding theory*. Cambridge: Cambridge University Press.

Rolls, E. T., & Deco, G. (2010). *The noisy brain: Stochastic dynamics as a principle of brain function*. New York: Oxford University Press.

Salavati, A. H., & Karbasi, A. (2012). Multi-level error-resilient neural networks. In *Proc. 2012 IEEE Int. Symp. Inf. Theory* (pp. 1064–1068). Piscataway, NJ: IEEE.

Sarpeshkar, R. (1998). Analog versus digital: Extrapolating from electronics to neurobiology. *Neural Comput.*, *10*(7), 1601–1638.

Shannon, C. E. (1948). A mathematical theory of communication. *Bell Syst. Tech. J.*, *27*, 379–423, 623–656.

Song, S., Sjöström, P. J., Reigl, M., Nelson, S., & Chklovskii, D. B. (2005). Highly nonrandom features of synaptic connectivity in local cortical circuits. *PLoS Biol.*, *3*(3), 0507–0519.

Spielman, D. A., & Teng, S.-H. (2004). Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *J. ACM*, *51*(3), 385–463.

Stettler, D. D., & Axel, R. (2009). Representations of odor in the piriform cortex. *Neuron*, *63*(6), 854–864.

Tabatabaei Yazdi, S. M. S., Cho, H., & Dolecek, L. (2013). Gallager B decoder on noisy hardware. *IEEE Trans. Commun.*, *61*(5), 1660–1673.

Taylor, M. G. (1968). Reliable information storage in memories designed from unreliable components. *Bell Syst. Tech. J.*, *47*(10), 2299–2337.

Treves, A., & Rolls, E. T. (1994). Computational analysis of the role of the hippocampus in memory. *Hippocampus*, *4*(3), 374–391.

Varshney, L. R. (2011). Performance of LDPC codes under faulty iterative decoding. *IEEE Trans. Inf. Theory*, *57*(7), 4427–4444.

Varshney, L. R., Sjöström, P., & Chklovskii, D. B. (2006). Optimal information storage in noisy synapses under resource constraints. *Neuron*, *52*(3), 409–423.

Venkatesh, S. S. (1994). Connectivity versus capacity in the Hebb rule. In V. Roychowdhury, K.-Y. Siu, & A. Orlitsky (Eds.), *Theoretical advances in neural computation and learning* (pp. 173–240). New York: Kluwer Academic.

Vincent, P., Larochelle, H., Bengio, Y. & Manzagol, P.-A. (2008). Extracting and composing robust features with denoising autoencoders. In *Proc. 25th Int. Conf. Mach. Learn. (ICML 2008)* (pp. 1096–1103). New York: ACM.

Wilson, D. A., & Sullivan, R. M. (2011). Cortical processing of odor objects. *Neuron*, *72*(4), 506–519.

Xu, L., Krzyzak, A., & Oja, E. (1991). Neural nets for dual subspace pattern recognition method. *Int. J. Neur. Syst.*, *2*(3), 169–184.

Yoshida, M., Hayashi, H., Tateno, K., & Ishizuka, S. (2002). Stochastic resonance in the hippocampal CA3–CA1 model: A possible memory recall mechanism. *Neural Netw.*, *15*(10), 1171–1183.

Received March 6, 2014; accepted May 15, 2014.